**1. JNI 실습**

NDK 설치를 통해 C++로 된 Native 코드를 컴파일 해서 수행하기 위해, JNI를 이용하여 Java class에서 사용하고자 할때의 과제이다.

0. 처음에 환경설정에서 환경변수에 path에 NDK를 넣어주는 과정부터 ppt에 있는 과정을 순서대로 따라 진행하였다.

프로젝트 이름 : testjni

1. JNIClass.java 를 app/java안에 만들어주었다

package com.example.testjni;  
  
public class JNIClass {  
 static  
 {  
 System.*loadLibrary*("jniExample");  
 }  
 public native String getNumString(String str, int num);  
  
 private String callback(String str, int num){  
  
 return str + " : " + num;  
 }  
}

2. MainActivity.java에 다음 코드를 적는다.

package com.example.testjni;  
  
import androidx.appcompat.app.AppCompatActivity;  
  
import android.os.Bundle;  
import android.view.View;  
import android.widget.Button;  
import android.widget.TextView;  
  
public class MainActivity extends AppCompatActivity {  
 JNIClass jni;  
 Button call;  
 TextView tv;  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
  
 jni = new JNIClass();  
  
 call = (Button)findViewById(R.id.*jniCall\_btn*);  
 tv = (TextView)findViewById(R.id.*jni\_txt*);  
 call.setOnClickListener(new View.OnClickListener(){  
 @Override  
 public void onClick(View view){  
 tv.setText(jni.getNumString("Native Method 호출", 1));  
 }  
 });  
 }  
}

3. activity\_main.xml에 다음 코드를 적는다

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 tools:context=".MainActivity">  
  
 <Button  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Jni Call"  
 android:id="@+id/jniCall\_btn"/>  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Hello World! 201524404 강민진"  
 android:id="@+id/jni\_txt"/>  
  
</LinearLayout>

4. .class파일을 만들어서 c/c++ 파일을 동작할 수 있게 한다.

빌드하고 해당 프로젝트 안에 class파일이 생성되었는지를 확인한다.

5. app/src/main안에 jni폴더를 만든다.

6. File -> Setting -> Tools -> External Tools를 통해 헤더파일을 만든다.

프로그램(javah.exe) 설정, 인자 설정(-classpath “$Classpath$” -v -jni $FileClass$로 설정)

Working Directory 를 내 프로젝트에 맞게 설정한다

그 후 NDK에 있는 javah를 클릭해서 헤더파일을 생성하고 이를 확인한다.

7.jni폴더 내에 JNIMain.cpp를 생성하고 다음 코드를 작성한다.

#include "com\_example\_testjni\_JNIClass.h"  
  
JNIEXPORT jstring JNICALL Java\_com\_example\_testjni\_JNIClass\_getNumString  
 (JNIEnv \* a, jobject b, jstring str, jint num)  
 {  
 jclass cls = a->GetObjectClass(b);  
 jmethodID func = a->GetMethodID(cls, "callback", "(Ljava/lang/String;l)Ljava/lang/String;");  
 return (jstring)a->CallObjectMethod(b, func, str, num);  
 }

8. jni폴더 내에 Android.mk파일을 생성하고 다음과 같은 코드를 작성한다.

LOCAL\_PATH := $(call my-dir)  
  
include $(CLEAR\_VARS)  
  
LOCAL\_MODULE := jniExample  
LOCAL\_SRC\_FILES := jniMain.cpp  
  
include $(BUILD\_SHARED\_LIBRARY)

9. javah를 만들었던 것과 동일하게 NDK내부에 ndk-build를 생성한다.

Program에는 ndk폴더

Parameter는 비우고

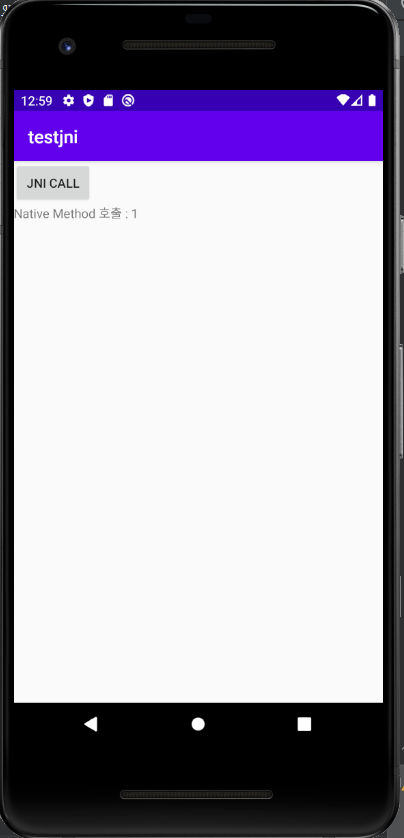
Working Directory 에는 jni폴더의 위치를 설정한다.

ndk 내부에 ndk-build 를 실행하고 libs가 적절한 경로에 생성되었는지를 확인한다.

10. Build-gradle 경로를 설정하고 Sync Now를 누른다.

apply plugin: 'com.android.application'  
  
android {  
 externalNativeBuild{  
 ndkBuild{  
 path "C:/Users/82104/AndroidStudioProjects/testjni/app/src/main/jni/Android.mk"  
 }  
 }  
 compileSdkVersion 29  
 buildToolsVersion "29.0.3"  
  
 defaultConfig {  
 applicationId "com.example.testjni"  
 minSdkVersion 29  
 targetSdkVersion 29  
 versionCode 1  
 versionName "1.0"  
  
 testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"  
 }  
  
 buildTypes {  
 release {  
 minifyEnabled false  
 proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'  
 }  
 }  
  
}  
  
dependencies {  
 implementation fileTree(dir: 'libs', include: ['\*.jar'])  
  
 implementation 'androidx.appcompat:appcompat:1.1.0'  
 implementation 'androidx.constraintlayout:constraintlayout:1.1.3'  
 testImplementation 'junit:junit:4.12'  
 androidTestImplementation 'androidx.test.ext:junit:1.1.1'  
 androidTestImplementation 'androidx.test.espresso:espresso-core:3.2.0'  
}

11. 그 후 실행한다.



실행 시 왼쪽에서 오른쪽으로 변경된 것을 확인할 수 있다.

**2. SQLite 실습**

DB는 상호 연관있는 데이터들이 모여있는 것이다. 여러 데이터를 구조화하여 저장하고 있다.

데이터의 중복을 최소화하고 검색 및 공유가 용이하도록 한다.

DBMS(DataBase Management System) : 사용자의 요구에 따라 정보를 생성해주고 DB를 관리하는 소프트웨어이다. 기존 파일 시스템에서의 데이터 종속성 및 중복성 문제를 해결하였다.

SQL은 DB를 이용 및 접근하기 위한 DBMS와의 통신수단, 언어이다.

데이터 정의어 및 데이터 조작어를 포함한다. DB에 대한 질의 기능을 갖고 있다.

SQLite : DBMS의 일종용량은 적지만 안정적임이며, Server용이 아닌 응용 프로그램에 내부적으로 주로 사용한다. 데이터를 하나의 파일에 정리하여 저장하며, C언어로 작성이 되어 처리 속도가 빠르고, 안드로이드에 기본 탑재된 DBMS이다.

프로젝트명 : sqltest

1. MainActivity가 있는 곳에 DBHelper.java를 생성해주고 다음 코드를 작성하였다.

package com.example.sqltest;  
import android.content.Context;  
import android.database.sqlite.SQLiteDatabase;  
import android.database.sqlite.SQLiteOpenHelper;  
  
public class DBHelper extends SQLiteOpenHelper {  
 public DBHelper(Context context, String name, SQLiteDatabase.CursorFactory factory, int version){  
 super(context, name, factory, version);  
 }  
 @Override  
 public void onCreate(SQLiteDatabase db){  
 String sql = "create table if not exists student("  
 + "\_id integer primary key autoincrement, "  
 + "name text, "  
 + "age integer, "  
 + "address text);";  
 db.execSQL(sql);  
 }  
 @Override  
 public void onUpgrade(SQLiteDatabase db, int odlVersion, int newVersion) {  
 String sql = "drop table if exists students";  
 db.execSQL(sql);  
 onCreate(db);  
 }  
}

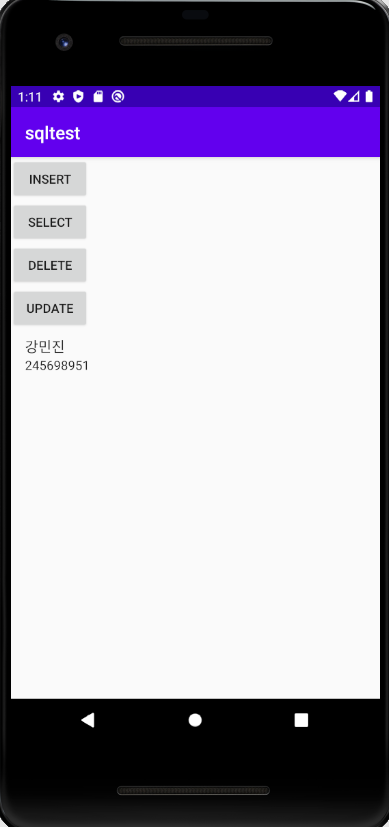
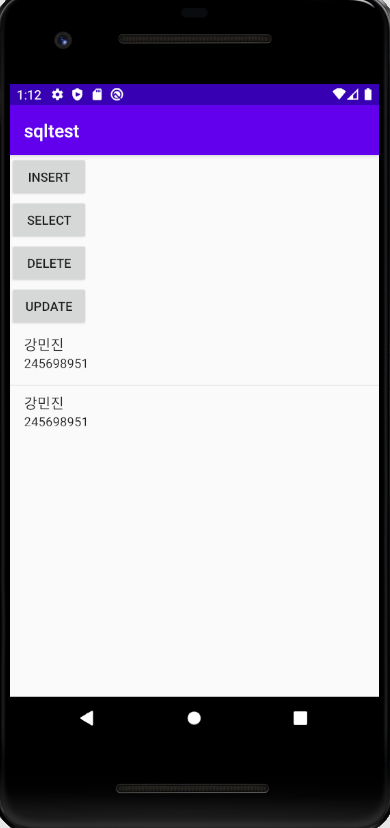
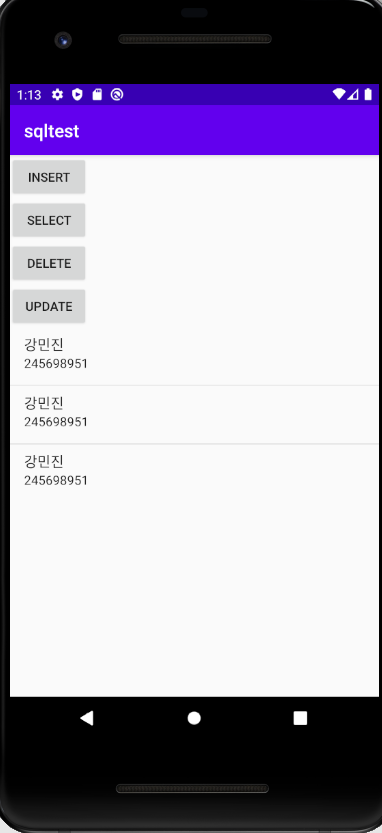
2. MainActivity.java에는 다음과 같은 코드를 작성하였다.(insert, delete, select 등 후의 작업 포함.)

package com.example.sqltest;  
  
import androidx.appcompat.app.AppCompatActivity;  
  
import android.content.ContentValues;  
import android.database.Cursor;  
import android.database.sqlite.SQLiteDatabase;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.Button;  
import android.widget.ListView;  
import android.widget.SimpleCursorAdapter;  
  
public class MainActivity extends AppCompatActivity {  
 DBHelper helper;  
 SQLiteDatabase db;  
 Button insert\_btn;  
 Button delete\_btn;  
 Button select\_btn;  
 Button update\_btn;  
   
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
 insert\_btn = (Button)findViewById(R.id.*insert\_btn*);  
 select\_btn = (Button)findViewById(R.id.*select\_btn*);  
 helper = new DBHelper(MainActivity.this,"person.db",null,1);  
 db = helper.getWritableDatabase();  
 helper.onCreate(db);  
 insert\_btn.setOnClickListener(new View.OnClickListener(){  
 @Override  
 public void onClick(View view){  
 ContentValues values = new ContentValues();  
  
 values.put("name","강민진");  
 values.put("age", 245698951);  
 values.put("address","부산시 금정구 장전동");  
 db.insert("student", null,values);  
 }  
 });  
 delete\_btn = (Button)findViewById(R.id.*delete\_btn*);  
 delete\_btn.setOnClickListener(new View.OnClickListener(){  
 @Override  
 public void onClick(View view){  
 db.delete("student", "name=?", new String[]{"강민진"});  
 }  
 });  
 update\_btn = (Button)findViewById(R.id.*update\_btn*);  
 update\_btn.setOnClickListener(new View.OnClickListener(){  
 @Override  
 public void onClick(View view){  
 ContentValues new\_values = new ContentValues();  
 new\_values.put("age", 201524404);  
 db.update("student", new\_values,"name=?",new String[]{"강민진"});  
 }  
 });  
 select\_btn.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 Cursor c = db.query("student",null,null,null,null,null,null,null);  
 SimpleCursorAdapter adapter = null;  
 adapter = new SimpleCursorAdapter(MainActivity.this,  
 android.R.layout.*simple\_list\_item\_2*, c,  
 new String[] {"name","age","address"},  
 new int[]{android.R.id.*text1*, android.R.id.*text2*}, 0);  
 ListView list = (ListView)findViewById(R.id.*list*);  
 list.setAdapter(adapter);  
 }  
 });  
 }  
}

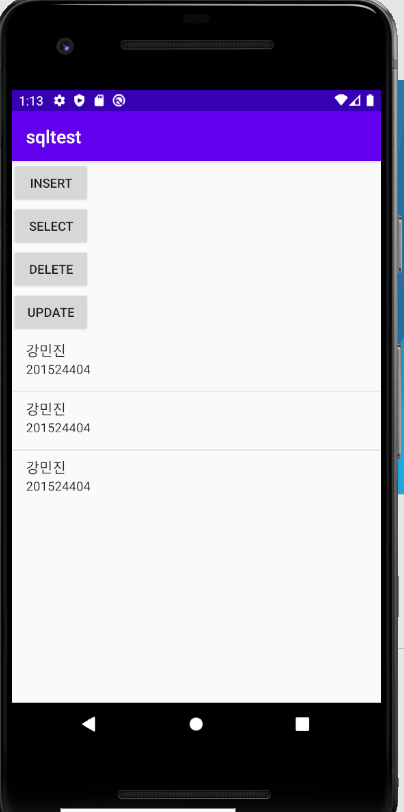
3. Activity\_main.xml layout에 다음과 같은 코드를 추가하였다(위와 마찬가지로 insert, delete, select등의 코드 작성)

처음에는 이상한 학번(age)이 나오다가 update시 제대로 된 학번이 나오게 구현하였다.

1) insert 후 select 이렇게 3번을 반복하였다.

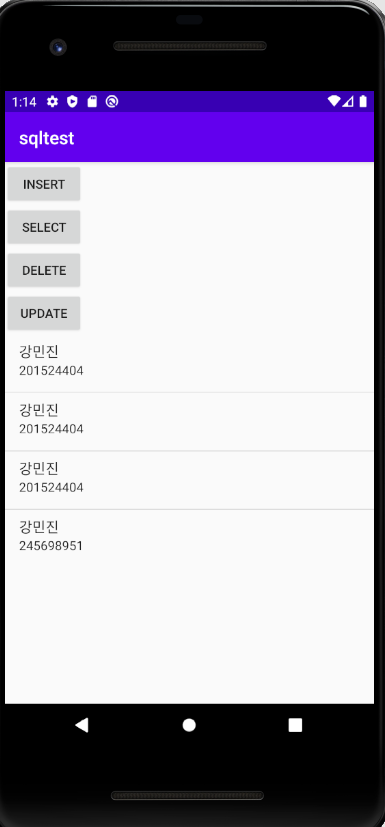
  

2) update후 select를 하였다



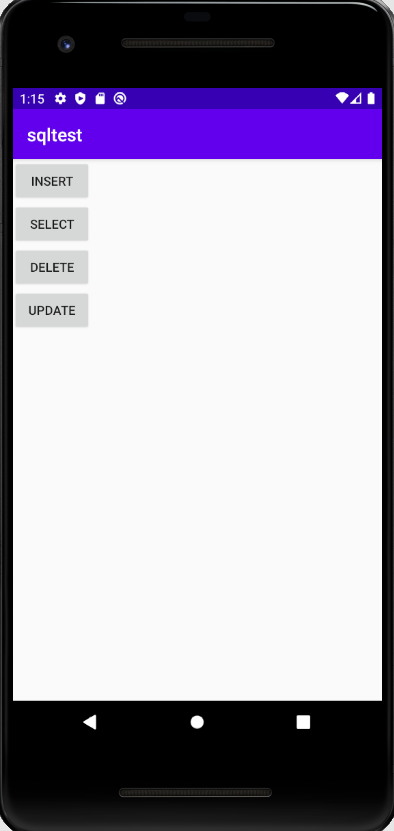
학번이 정확하게 출력된 것을 확인할 수 있다.

3) insert후 select를 다시 하였다.



아까와 동일하게 update되지 않은 학번이 새로 insert 된것에서 나오는 것을 확인할 수 있다.

4) delete후 select를 하였다.



지워진 것을 확인할 수 있다.

결론 및 느낀점)

1번 JNI과제를 수행할때는 정말 많이 헤맸다.

코드를 작성할때 잘못 작성한 부분이 있었는데 그런 부분에 대한 오류 처리가 안되어서 몇번을 처음부터 다시 했다.

버전의 문제인지는 모르겠으나 중간에 오류가 많이 생성되었는데 그때마다 처음부터 다시 만들어주는 번거로움이 생겼다. 환경을 만들거나 새로운 것을 도입할 때는 서두르지 말고 한단계씩 차근차근 밟아야 겠다는 것을 느꼈다.

또한 ppt의 프로젝트 위치, ndk위치 등이 내 컴퓨터의 것과 달라서 Working Directory 설정, 일일이 찾아줘야 하는 과정이 매우 번거로웠다. ppt를 참고하고, 인터넷을 참고하면서 포기하지 않고 하다보니 해결을 할 수 있었다.

2번 SQLite과제는 1번과제보다 매우 수월했다. 그저 ppt에 있는 과정을 그대로 진행하다 보니 완성되었다.

JAVA에 C/C++코드를 사용할 수 있다는 것을 배웠으며, DB와 DB를 컨트롤할 수 있는 SQL, SQLite에 대하여 배웠다. 또한 이 과제를 통해서 부수적인 여러 역량들이 많이 늘었다고 느꼈다.